ISLAMIC UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering (CSE)
Course Outline and Course Plan

Name of the Teacher	**S. M. Sabit Bananee**	**Position**	**Lecturer**
Department	**CSE**	**Programme**	**B.Sc. Eng. in SWE**
Course Code	**CSE 4501**	**Course Title**	**Operating Systems**
Academic Year	**2021-22**	**Semester**	**Winter**
Contact Hours	**3.0**	**Credit Hours**	**3.0**
Text books and Reference books	1. ***Operating Systems Concepts, 9TH Ed*** 2. ***Understanding Operating System, 6TH Ed*** 3. ***Operating Systems: A Spiral Approach*** 4. ***Operating Systems: Internal Design and Principles, Pearson, 8th Ed*** 5. ***Modern Operating Systems, 4th Ed***	**Authors of the books**	**1. Silberschatz, A., Galvin, A., Gagne,** **2. McHoes, A., Flynn, I.** **3. Elmasri, R., Carrick, A., Levine, D.** **4. William Stallings** **5. Tanenbaum A. S.**
Prerequisites (If any)	1. **CSE 4303: Data Structures** 2. **CSE 4305: Computer Organization and Architecture**		
Course Homepage	https://classroom.google.com/c/NTI2MTg5OTE4NTg0		
Teaching Methods/ Approaches	☐ **Lecture√** ☐ **Project**	☐ **Group discussion√** ☐ **Others: Tutorial classes√**	☐ **Demonstration** ☐ **Problem solving√**
Teaching aids	**Multi-media**	**OHP**	**Board and Marker√** **Others**

Course Assessment Method								
Attendance (10%)	**Quiz 30% of Total Marks (Best 3 out of 4)**						**Mid Semester (25%)**	**Semester Final (35%)**
	1st Quiz	**2nd Quiz**	**3rd Quiz**	**4th Quiz**	**Others**		**Week/Date**	**Week/Date**
	Week/Date	**Week/Date**	**Week/Date**	**Week/Date**	**Assignment**	**Homework**		
	4th Week	**6th Week**	**10th Week**	**13th Week**	**2 Assignments**	**Will be given time to time**	**As per schedule of IUT**	**As per schedule of IUT**

Grading Policy					
Marks out of 100	**Letter Grade**	**Grade Point**	**Marks out of 100**	**Letter Grade**	**Grade Point**
80 - 100	A+	4.00	55 - 59	B-	2.75
75 - 79	A	3.75	50 - 54	C+	2.50
70 - 74	A-	3.50	45 - 49	C	2.25
65 - 69	B+	3.25	40 - 44	D	2.00
60 - 64	B	3.00	00 - 39	F	0.00

Class Schedule	
Wednesday	09:15 AM – 10:30 AM
Thursday	03:45 PM – 05:00 PM

Student’s consulting hour	Tuesday 2:30 PM to 3:00 PM

Course Contents
Operating System Overview, Unix / Windows History, POSIX, GNU / GLP, Homebrew Club, Open-Source OS Linux. Process Description and Control 2 and 5 states process models, Process Control Structures, Modes of Execution, Process Switching Threads User-Level and Kernel-Level Threads, Performance on Multicore, Linux Process and Thread Management Concurrency: Mutual Exclusion and Synchronization Race Condition, Interrupt Disabling, Producer/Consumer Problem, Monitors, Message Passing Concurrency: Deadlock and Starvation Principles of Deadlock, Hold and Wait, Circular wait, Deadlock Detection Algorithm Memory Management Relocation, Protection, Memory Partitioning, Paging, Segmentation Scheduling Types of Processor Scheduling, Scheduling Algorithms, Traditional UNIX Scheduling I/O Management DMA, I/O Buffering, Disk Scheduling, UNIX SVR4 I/O, Linux I/O OS Security & Threats, Threats, Attacks, and Assets, Malicious Software, Viruses, Worms, and Bots, Rootkits

Course Objectives

After completing the course, the student must be able to:

- ✓ Explain what operating systems are, what they do, and how they are designed and constructed;
- ✓ Discuss various methods for process management and CPU scheduling;
- ✓ Explain the principals involved in the internal algorithms and structures of primary and secondary memory management.
- ✓ Identify and discuss the protection mechanisms that may be provided by operating systems.

Mapping with CO, PO and Bloom’s Taxonomy

CO No.	Course Outcomes (CO) Statement	levels of Bloom’s Taxonomy	Matching with Program Outcome (PO)
CO1	Describe the evolution, types, structure and functions of operating systems	C2	PO1, PO2
CO2	Explain techniques involved in process, memory, device and file management	C2	PO2
CO3	Describe security and protection measures used in operating systems	C2	PO3
CO4	Execute Linux basic commands and shell scripts	C3	PO2
CO5	Implement processor scheduling, synchronization, deadlocks and disk allocation algorithms for a given scenario	C3	PO1, PO2

Weekly plan for course content and mapping with CO

Weeks	Topics	COs
1	Class orientation Discussion of course goals, expected outcomes, course policies and grading system	---
2	Introduction to Operating Systems	CO1
3 & 4	Computer System Structures	CO1, CO2
5	Process Management	CO2, CO3
6 & 7	CPU Scheduling	CO5

8 & 9	**MID TERM EXAMINATIONS**	
10 & 11	Virtual Memory	CO2
12 & 13	Memory Management	CO2
14 & 15	Disk Scheduling	CO5
16	Case Study of Different Platforms • MS-DOS • MAC-OS • UNIX/LINUX • IBM MVS/DOS • ANDROID • SYMBIAN, etc	CO4, CO5
17 - 19	**FINAL EXAMINATIONS**	

Mapping of Course Outcomes (COs) and Program Outcomes (POs) and Evaluation Methods						
Assessment Method	**Marks**	Mark distributions (as %) on COs and POs				
		CO1	CO2	CO3	CO4	CO5
		PO1 PO2	PO2	PO3	PO2	PO1 PO2
Attendance (Class Participation)	10%	-	-	-	-	-
Quiz 1/Quiz 2/Quiz 3/Quiz 4	15%	5%	5%	-	-	5%
Midterm Exam.	25%	8.33%	8.33%	-	-	8.33%
Final Exam.	50%	5%	5%	10%	10%	20%
Total	100%	18.33%	18.33%	10%	10%	33.33%

Mapping of COs and POs [Correlation level 1 for low, 2 for moderate and 3 for high]												
Course Outcomes	**PO1**	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**
CO1	√	√										
CO2		√										
CO3	√											
CO4		√										
CO5	√	√										

Program Outcomes	
PO 1	**Engineering Knowledge:** Apply knowledge of **mathematics, natural science**, **engineering** fundamentals and system fundamentals, software development, networking & communication, and information assurance & security to the solution of complex engineering problems in computer science and engineering.
PO 2	**Problem Analysis:** Ability to **identify**, **formulate** and **analyze complex** Computer Science and Engineering problems in the areas of hardware, software, theoretical Computer Science and applications to reach significant conclusions by applying Mathematics, Natural sciences, Computer Science and Engineering principles.
PO 3	**Design/ Development of Solutions:** **Design solutions** for complex computer science and engineering problems and design systems, components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
PO 4	**Investigation:** Ability to use **research-based knowledge** and **research methods** to perform literature survey, design experiments for complex problems in designing, developing and maintaining a computing system, collect data from the experimental outcome, analyze and interpret valid/interesting patterns and conclusions from the data points.
PO 5	**Modern Tool Usage:** Ability to create, select and apply **state of the art tools** and techniques in designing, developing and testing a computing system or its component.
PO 6	**The Engineer and Society:** Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to **professional engineering practice** in system development and solutions to **complex engineering problems** related to system fundamentals, software development, networking & communication, and information assurance & security.
PO 7	**Environment and Sustainability:** Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to **professional engineering practice** in system development and solutions to **complex engineering problems** related to system fundamentals, software development, networking & communication, and information assurance & security.
PO 8	**Ethics:** Apply **ethical principles** and commit to **professional ethics** and **responsibilities** and norms of computer science and engineering practice.
PO 9	**Individual Work and Teamwork:** Ability to function as an individual and as a team player or leader in multidisciplinary teams and strive towards **achieving a common goal.**
PO 10	**Communication:** **Communicate effectively** on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO 11	**Project Management and Finance:** Demonstrate knowledge and understanding of engineering management principles and economic decision making and apply these to one’s own work, as a member and leader in a team, to **manage projects** and in multidisciplinary environments.
PO 12	**Life-long learning:** Recognize the need for, and have the preparation and ability to **engage in independent** and **lifelong learning** in the broadest context of technological change.

K

Table 4.1: Knowledge Profile

	Attribute
K1	A systematic, theory-based understanding of the natural sciences applicable to the discipline
K2	Conceptually based mathematics, numerical analysis, statistics and the formal aspects of computer and information science to support analysis and modeling applicable to the discipline
K3	A systematic, theory-based formulation of engineering fundamentals required in the engineering discipline
K4	Engineering specialist knowledge that provides theoretical frameworks and bodies of knowledge for the accepted practice areas in the engineering discipline; much is at the forefront of the discipline
K5	Knowledge that supports engineering design in a practice area
K6	Knowledge of engineering practice (technology) in the practice areas in the engineering discipline
K7	Comprehension of the role of engineering in society and identified issues in engineering practice in the discipline: ethics and the engineer's professional responsibility to public safety; the impacts of engineering activity; economic, social, cultural, environmental and sustainability
K8	Engagement with selected knowledge in the research literature of the discipline

P

Table 4.2: Range of Complex Engineering Problem Solving

Attribute	**Complex Engineering Problems** have characteristic P1 and some or all of P2 to P7:
Depth of knowledge required	P1: Cannot be resolved without in-depth engineering knowledge at the level of one or more of K3, K4, K5, K6 or K8 which allows a fundamentals-based, first principles analytical approach
Range of conflicting requirements	P2: Involve wide-ranging or conflicting technical, engineering and other issues
Depth of analysis required	P3: Have no obvious solution and require abstract thinking, originality in analysis to formulate suitable models
Familiarity of issues	P4: Involve infrequently encountered issues
Extent of applicable codes	P5: Are outside problems encompassed by standards and codes of practice for professional engineering
Extent of stakeholder involvement and conflicting requirements	P6: Involve diverse groups of stakeholders with widely varying needs
Interdependence	P7: Are high level problems including many component parts or sub-problems

A

Table 4.3: Range of Complex Engineering Activities

Attribute	**Complex activities** means (engineering) activities or projects that have some or all of the following characteristics:
Range of resources	A1: Involve the use of diverse resources (and for this purpose resources include people, money, equipment, materials, information and technologies)
Level of interaction	A2: Require resolution of significant problems arising from interactions between wide-ranging or conflicting technical, engineering or other issues
Innovation	A3: Involve creative use of engineering principles and research-based knowledge in novel ways
Consequences for society and the environment	A4: Have significant consequences in a range of contexts, characterized by difficulty of prediction and mitigation
Familiarity	A5: Can extend beyond previous experiences by applying principles-based approaches

Instructor contact details:
S. M. Sabit Bananee
Lecturer
Department of Computer Science and Engineering
Islamic University of Technology
Room No: 602, Academic Building-2, Mobile: 01925215306
Email: smsabitbananee@iut-dhaka.edu

A machine learning algorithm based on decision trees is called Random Forest Trees (RFT). Machine learning algorithms that do ensemble classification include Random Trees (RT). The term "ensemble" denotes a technique that averages the forecasts of various different base models to provide predictions.
The core idea behind ensemble methods based on randomization is to "incorporate random perturbations into the learning procedure to build multiple alternative models from a single learning set L and then to aggregate the predictions of those models to make the ensemble prediction" (Louppe, 2014). In other words, "growing an ensemble of trees and letting them vote for the most popular class has resulted in significant gains in classification accuracy. These ensembles are frequently grown by creating random vectors that control how each tree in the ensemble grows (Breiman, 2001).

When building a random tree, there are three basic options available. These three considerations are: (1) how to separate leaves; (2) what kind of predictor to utilize in each leaf; and (3) how to introduce unpredictability into trees (Denil et al., 2014). Using a bootstrapped or sub-sampled data set to generate each tree is a typical method for adding unpredictability to a tree. As a result, there are variances among the trees in the forest since each tree in the forest was trained using slightly different data (Denil et al., 2014). The optimal split at a particular node can alternatively be chosen randomly; tests have shown, however, that where noise is relevant, bagging typically produces better results (Louppe, 2014).

"Special attention must be taken so that the resulting model is neither too simple nor too complex," according to the author, when optimizing a Random Trees model. The model is in fact stated to have underfitted the data in the first scenario, i.e., it was not adaptable enough to capture the structure between X and Y. The model is said to be overfit the data in the latter scenario because it is too flexible and captures isolated structures (i.e., noise) that are unique to the learning set (Louppe, 2014).

In order to prevent overfitting, stropping rules must be established to stop a tree from developing before it has too many levels: User-defined hyper-parameters are used to establish stopping conditions (Louppe, 2014). The most popular of these parameters are:
The bare minimum of samples that a terminal node needs to divide
the bare minimum of samples in a leaf node after splitting the terminal node
The maximum depth of a tree, or the number of levels it can reach,
once the Gini Impurity index, which measures the Trees accuracy, falls below a predetermined threshold
To identify the best trade-off, these parameters must be fine-tuned; they must be neither too stringent nor too loose for the tree to be neither too shallow nor too deep (Louppe, 2014).
Breiman (2002) lists the following as some of the essential characteristics of random trees:
It is a very good classifier, with accuracy on par with support vector machines.
As the forest grows, it produces an internal, unbiased estimate of the generalization error.
When up to 80% of the data are missing, it nevertheless retains accuracy thanks to an efficient estimation algorithm.
It has a technique for balancing inaccuracy in data sets with an imbalanced class population.

The generated forests can be saved for use on other data in the future.
It provides an estimate of the variables that are crucial for classification.
Information regarding the relationship between the variables and the categorization is shown in the output that is produced.
It calculates distances between examples that can be used for grouping, finding outliers, or scaling to provide intriguing data visualizations.
Contrary to the Support Vector Machine (SVM), the random trees classifier can typically handle a mix of categorical and numerical variables. As for data scaling, Random Trees are less susceptible to it than SVM, which frequently requires data to be normalized before training or classification. SVM is said to perform better, nonetheless, when the training set is little or uneven. Comparable in computational complexity to SVM, the Random Trees classifier performs better and more quickly with big training sets.